Network Working Group	D. Harrington
Internet-Draft	Huawei Technologies (USA)
Intended status: Standards Track	W. Hardaker
Expires: November 7, 2009	Sparta, Inc.
	May 06, 2009

TOC

Transport Security Model for SNMP
draft-ietf-isms-transport-security-model-14

Status of This Memo

Copyright Notice

Abstract

This memo describes a Transport Security Model for the Simple Network
Management Protocol.
This memo also defines a portion of the Management Information Base
(MIB) for monitoring and managing the Transport Security Model for
SNMP.

Table of Contents

---

1. Introduction

TOC

This memo describes a Transport Security Model for the Simple Network Management Protocol, for use with secure Transport Models in the Transport Subsystem [I-D.ietf-isms-tmsm] (Harrington, D. and J. Schoenwaelder, “Transport Subsystem for the Simple Network Management Protocol (SNMP),” May 2009.).
This memo also defines a portion of the Management Information Base (MIB) for monitoring and managing the Transport Security Model for SNMP.
It is important to understand the SNMP architecture and the terminology of the architecture to understand where the Transport Security Model described in this memo fits into the architecture and interacts with other subsystems and models within the architecture. It is expected that reader will have also read and understood RFC3411 [RFC3411] (Harrington, D., Presuhn, R., and B. Wijnen, “An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks,” December 2002.), RFC3412 [RFC3412] (Case, J., Harrington, D., Presuhn, R., and B. Wijnen, “Message Processing and Dispatching for the Simple Network Management Protocol (SNMP),” December 2002.), RFC3413 [RFC3413] (Levi, D., Meyer, P., and B. Stewart, “Simple Network Management Protocol (SNMP) Applications,” December 2002.), and RFC3418 [RFC3418] (Presuhn, R., “Management Information Base (MIB) for the Simple Network Management Protocol (SNMP),” December 2002.).

1.1. The Internet-Standard Management Framework

TOC

For a detailed overview of the documents that describe the current Internet-Standard Management Framework, please refer to section 7 of RFC 3410 [RFC3410] (Case, J., Mundy, R., Partain, D., and B. Stewart,

"Introduction and Applicability Statements for Internet-Standard Management Framework," December 2002.).
Managed objects are accessed via a virtual information store, termed the Management Information Base or MIB. MIB objects are generally accessed through the Simple Network Management Protocol (SNMP). Objects in the MIB are defined using the mechanisms defined in the Structure of Management Information (SMI). This memo specifies a MIB module that is compliant to the SMIv2, which is described in STD 58, RFC 2578 [RFC2578] (McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)," April 1999.), STD 58, RFC 2579 [RFC2579] (McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2," April 1999.) and STD 58, RFC 2580 [RFC2580] (McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2," April 1999.).

1.2. Conventions

TOC

For consistency with SNMP-related specifications, this document favors terminology as defined in STD62 rather than favoring terminology that is consistent with non-SNMP specifications that use different variations of the same terminology. This is consistent with the IESG decision to not require the SNMPv3 terminology be modified to match the usage of other non-SNMP specifications when SNMPv3 was advanced to Full Standard.
Authentication in this document typically refers to the English meaning of "serving to prove the authenticity of" the message, not data source authentication or peer identity authentication.
The terms "manager" and "agent" are not used in this document, because in the RFC 3411 architecture, all SNMP entities have the capability of acting as either manager or agent or both depending on the SNMP applications included in the engine. Where distinction is needed, the application names of Command Generator, Command Responder, Notification Originator, Notification Receiver, and Proxy Forwarder are used. See "SNMP Applications" [RFC3413] (Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications," December 2002.) for further information.
While security protocols frequently refer to a user, the terminology used in RFC3411 [RFC3411] (Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks," December 2002.) and in this memo is "principal". A principal is the "who" on whose behalf services are provided or processing takes place. A principal can be, among other things, an individual acting in a particular role; a set of individuals, with each acting in a particular role; an application or a

set of applications, or a combination of these within an administrative domain.
The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.).
Non uppercased versions of the keywords should be read as in normal English. They will usually, but not always, be used in a context relating to compatibility with the RFC3411 architecture or the subsystem defined here, but which might have no impact on on-the-wire compatibility. These terms are used as guidance for designers of proposed IETF models to make the designs compatible with RFC3411 subsystems and Abstract Service Interfaces (ASIs) (see section 3.2). Implementers are free to implement differently. Some usages of these lowercase terms are simply normal English usage.

1.3. Modularity

TOC

The reader is expected to have read and understood the description of the SNMP architecture, as defined in [RFC3411] (Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks," December 2002.), and the architecture extension specified in "Transport Subsystem for the Simple Network Management Protocol" [I-D.ietf-isms-tmsm] (Harrington, D. and J. Schoenwaelder, "Transport Subsystem for the Simple Network Management Protocol (SNMP)," May 2009.), which enables the use of external "lower layer transport" protocols to provide message security, tied into the SNMP architecture through the Transport Subsystem. The Transport Security Model is designed to work with such lower-layer secure Transport Models.
In keeping with the RFC 3411 design decisions to use self-contained documents, this memo includes the elements of procedure plus associated MIB objects which are needed for processing the Transport Security Model for SNMP. These MIB objects SHOULD NOT be referenced in other documents. This allows the Transport Security Model to be designed and documented as independent and self-contained, having no direct impact on other modules, and allowing this module to be upgraded and supplemented as the need arises, and to move along the standards track on different time-lines from other modules.
This modularity of specification is not meant to be interpreted as imposing any specific requirements on implementation.

TOC

1.4. Motivation

This memo describes a Security Model to make use of Transport Models that use lower layer secure transports and existing and commonly deployed security infrastructures. This Security Model is designed to meet the security and operational needs of network administrators, maximize usability in operational environments to achieve high deployment success and at the same time minimize implementation and deployment costs to minimize the time until deployment is possible.

1.5. Constraints

TOC

The design of this SNMP Security Model is also influenced by the following constraints:

1. In times of network stress, the security protocol and its underlying security mechanisms SHOULD NOT depend solely upon the ready availability of other network services (e.g., Network Time Protocol (NTP) or Authentication, Authorization, and Accounting (AAA) protocols).

2. When the network is not under stress, the Security Model and its underlying security mechanisms MAY depend upon the ready availability of other network services.

3. It might not be possible for the Security Model to determine when the network is under stress.

4. A Security Model SHOULD NOT require changes to the SNMP architecture.

5. A Security Model SHOULD NOT require changes to the underlying security protocol.

---

2. How the Transport Security Model Fits in the Architecture

TOC

The Transport Security Model is designed to fit into the RFC3411 architecture as a Security Model in the Security Subsystem, and to utilize the services of a secure Transport Model.
For incoming messages, a secure Transport Model will pass a tmStateReference cache, described later. To maintain RFC3411 modularity, the Transport Model will not know which securityModel will process the incoming message; the Message Processing Model will

determine this. If the Transport Security Model is used with a non-secure Transport Model, then the cache will not exist or not be populated with security parameters, which will cause the Transport Security Model to return an error (see section 5.2)
The Transport Security Model will create the securityName and securityLevel to be passed to applications, and verify that the tmTransportSecurityLevel reported by the Transport Model is at least as strong as the securityLevel requested by the Message Processing Model.
For outgoing messages, the Transport Security Model will create a tmStateReference cache (or use an existing one), and pass the tmStateReference to the specified Transport Model.

2.1. Security Capabilities of this Model

TOC

2.1.1. Threats

TOC

The Transport Security Model is compatible with the RFC3411 architecture, and provides protection against the threats identified by the RFC 3411 architecture. However, the Transport Security Model does not provide security mechanisms such as authentication and encryption itself. Which threats are addressed and how they are mitigated depends on the Transport Model used. To avoid creating potential security vulnerabilities, operators should configure their system so this Security Model is always used with a Transport Model that provides appropriate security, where "appropriate" for a particular deployment is an administrative decision.

2.1.2. Security Levels

TOC

The RFC 3411 architecture recognizes three levels of security:

- without authentication and without privacy (noAuthNoPriv)
- with authentication but without privacy (authNoPriv)
- with authentication and with privacy (authPriv)

The model-independent securityLevel parameter is used to request specific levels of security for outgoing messages, and to assert that specific levels of security were applied during the transport and processing of incoming messages.

The transport layer algorithms used to provide security should not be exposed to the Transport Security Model, as the Transport Security Model has no mechanisms by which it can test whether an assertion made by a Transport Model is accurate.
The Transport Security Model trusts that the underlying secure transport connection has been properly configured to support security characteristics at least as strong as reported in tmTransportSecurityLevel.

2.2. Transport Sessions

TOC

The Transport Security Model does not work with transport sessions directly. Instead the transport-related state is associated with a unique combination of transportDomain, transportAddress, securityName and securityLevel, and referenced via the tmStateReference parameter. How and if this is mapped to a particular transport or channel is the responsibility of the Transport Subsystem.

2.3. Coexistence

TOC

In the RFC3411 architecture, a Message Processing Model determines which Security Model SHALL be called. As of this writing, IANA has registered four Message Processing Models (SNMPv1, SNMPv2c, SNMPv2u/ SNMPv2*, and SNMPv3) and three other Security Models (SNMPv1, SNMPv2c, and the User-based Security Model).

2.3.1. Coexistence with Message Processing Models

TOC

The SNMPv1 and SNMPv2c message processing described in RFC3584 (BCP 74) [RFC3584] (Frye, R., Levi, D., Routhier, S., and B. Wijnen, “Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework,” August 2003.) always selects the SNMPv1(1) and SNMPv2c(2) Security Models. Since there is no mechanism defined in RFC3584 to select an alternative Security Model, SNMPv1 and SNMPv2c messages cannot use the Transport Security Model. Messages might still be able to be conveyed over a secure transport protocol, but the Transport Security Model will not be invoked.
The SNMPv2u/SNMPv2* Message Processing Model is a historic artifact for which there is no existing IETF specification.
The SNMPv3 message processing defined in RFC3412 [RFC3412] (Case, J., Harrington, D., Presuhn, R., and B. Wijnen, “Message Processing and

Dispatching for the Simple Network Management Protocol (SNMP)," December 2002.), extracts the securityModel from the msgSecurityModel field of an incoming SNMPv3Message. When this value is transportSecurityModel(YY), security processing is directed to the Transport Security Model. For an outgoing message to be secured using the Transport Security Model, the application MUST specify a securityModel parameter value of transportSecurityModel(YY) in the sendPdu Application Service Interface (ASI).
[-- NOTE to RFC editor: replace YY with actual IANA-assigned number, and remove this note.]

2.3.2. Coexistence with Other Security Models

TOC

The Transport Security Model uses its own MIB module for processing to maintain independence from other Security Models. This allows the Transport Security Model to coexist with other Security Models, such as the User-based Security Model [RFC3414] (Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)," December 2002.).

2.3.3. Coexistence with Transport Models

TOC

The Transport Security Model MAY work with multiple Transport Models, but the RFC3411 application service interfaces (ASIs) do not carry a value for the Transport Model. The MIB module defined in this memo allows an administrator to configure whether or not TSM prepends a transport model prefix to the securityName. This will allow SNMP applications to consider transport model as a factor when making decisions, such as access control, notification generation, and proxy forwarding.
To have SNMP properly utilize the security services coordinated by the Transport Security Model, this Security Model MUST only be used with Transport Models that know how to process a tmStateReference, such as the Secure Shell Transport Model [I-D.ietf-isms-secshell] (Harrington, D., Salowey, J., and W. Hardaker, "Secure Shell Transport Model for SNMP," May 2009.).

---

3. Cached Information and References

TOC

When performing SNMP processing, there are two levels of state information that might need to be retained: the immediate state linking

a request-response pair, and potentially longer-term state relating to transport and security. "Transport Subsystem for the Simple Network Management Protocol" (Harrington, D. and J. Schoenwaelder, “Transport Subsystem for the Simple Network Management Protocol (SNMP),” May 2009.) [I-D.ietf-isms-tmsm] defines general requirements for caches and references.
This document defines additional cache requirements related to the Transport Security Model.

3.1. Transport Security Model Cached Information

TOC

The Transport Security Model has specific responsibilities regarding the cached information.

3.1.1. securityStateReference

TOC

The Transport Security Model adds the tmStateReference received from the processIncomingMsg ASI to the securityStateReference. This tmStateReference can then be retrieved during the generateResponseMsg ASI, so that it can be passed back to the Transport Model.

3.1.2. tmStateReference

TOC

For outgoing messages, the Transport Security Model uses parameters provided by the SNMP application to lookup or create a tmStateReference.
For the Transport Security Model, the security parameters used for a response MUST be the same as those used for the corresponding request. This security model uses the tmStateReference stored as part of the securityStateReference when appropriate. For responses and reports, this security model sets the tmSameSecurity flag to true in the tmStateReference before passing it to a transport model.
For incoming messages, the Transport Security Model uses parameters provided in the tmStateReference cache to establish a securityName, and to verify adequate security levels.

TOC

3.1.3. Prefixes and securityNames

The SNMP-VIEW-BASED-ACM-MIB [RFC3415] (Wijnen, B., Presuhn, R., and K. McCloghrie, “View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP),” December 2002.), the SNMP-TARGET-MIB module [RFC3413] (Levi, D., Meyer, P., and B. Stewart, “Simple Network Management Protocol (SNMP) Applications,” December 2002.), and other MIB modules contain objects to configure security parameters for use by applications such as access control, notification generation, and proxy forwarding.
Transport domains and their corresponding prefixes are coordinated via the IANA registry "SNMP Transport Domains".
If snmpTsmConfigurationUsePrefix is set to true then all securityNames provided by, or provided to, the Transport Security Model MUST include a valid transport domain prefix.
If snmpTsmConfigurationUsePrefix is set to false then all securityNames provided by, or provided to, the Transport Security Model MUST NOT include a transport domain prefix.
The tmSecurityName in the tmStateReference stored as part of the securityStateReference does not contain a prefix.

---

4. Processing an Outgoing Message

TOC

An error indication might return an OID and value for an incremented counter and a value for securityLevel, and values for contextEngineID and contextName for the counter, and the securityStateReference if the information is available at the point where the error is detected.

4.1. Security Processing for an Outgoing Message

TOC

This section describes the procedure followed by the Transport Security Model.
The parameters needed for generating a message are supplied to the Security Model by the Message Processing Model via the generateRequestMsg() or the generateResponseMsg() ASI. The Transport Subsystem architectural extension has added the transportDomain, transportAddress, and tmStateReference parameters to the original RFC3411 ASIs.

```
  statusInformation =                -- success or errorIndication
        generateRequestMsg(
        IN   messageProcessingModel  -- typically, SNMP version
        IN   globalData              -- message header, admin data
        IN   maxMessageSize          -- of the sending SNMP entity
        IN   transportDomain         -- (NEW) specified by application
        IN   transportAddress        -- (NEW) specified by application
        IN   securityModel           -- for the outgoing message
        IN   securityEngineID        -- authoritative SNMP entity
        IN   securityName            -- on behalf of this principal
        IN   securityLevel           -- Level of Security requested
        IN   scopedPDU               -- message (plaintext) payload
        OUT  securityParameters      -- filled in by Security Module
        OUT  wholeMsg                -- complete generated message
        OUT  wholeMsgLength          -- length of generated message
        OUT  tmStateReference        -- (NEW)  transport info
             )


  statusInformation = -- success or errorIndication
        generateResponseMsg(
        IN   messageProcessingModel  -- typically, SNMP version
        IN   globalData              -- message header, admin data
        IN   maxMessageSize          -- of the sending SNMP entity
        IN   transportDomain         -- (NEW) specified by application
        IN   transportAddress        -- (NEW) specified by application
        IN   securityModel           -- for the outgoing message
        IN   securityEngineID        -- authoritative SNMP entity
        IN   securityName            -- on behalf of this principal
        IN   securityLevel           -- Level of Security requested
        IN   scopedPDU               -- message (plaintext) payload
        IN   securityStateReference  -- reference to security state
                                     -- information from original
                                     -- request
        OUT  securityParameters      -- filled in by Security Module
        OUT  wholeMsg                -- complete generated message
        OUT  wholeMsgLength          -- length of generated message
        OUT  tmStateReference        -- (NEW) transport info
             )
```

4.2. Elements of Procedure for Outgoing Messages

TOC

1) If there is a securityStateReference (Response or Report message), then this security model uses the cached information rather than the information provided by the ASI. Extract the tmStateReference from the

securityStateReference cache. Set the tmRequestedSecurityLevel to the value of the extracted tmTransportSecurityLevel. Set the tmSameSecurity parameter in the tmStateReference cache to true. The cachedSecurityData for this message can now be discarded.
2) If there is no securityStateReference (e.g., a Request-type or Notification message) then create a tmStateReference cache. Set tmTransportDomain to the value of transportDomain, tmTransportAddress to the value of transportAddress, and tmRequestedSecurityLevel to the value of securityLevel. (Implementers might optimize by pointing to saved copies of these session-specific values.) Set the transaction-specific tmSameSecurity parameter to false.
If the snmpTsmConfigurationUsePrefix object is set to false, then set tmSecurityName to the value of securityName.
If the snmpTsmConfigurationUsePrefix object is set to true, then use the transportDomain to look up the corresponding prefix. (Since the securityStateReference stores the tmStateReference with the tmSecurityName for the incoming message, and tmSecurityName never has a prefix, the prefix stripping step only occurs when we are not using the securityStateReference).

> If the prefix lookup fails for any reason, then the snmpTsmUnknownPrefixes counter is incremented, an error indication is returned to the calling module, and message processing stops.
>
> If the lookup succeeds, but there is no prefix in the securityName, or the prefix returned does not match the prefix in the securityName, or the length of the prefix is less than 1 or greater than four US-ASCII alpha-numeric characters, then the snmpTsmInvalidPrefixes counter is incremented, an error indication is returned to the calling module, and message processing stops.
>
> Strip the transport-specific prefix and trailing ':' character (US-ASCII 0x3a) from the securityName. Set tmSecurityName to the value of securityName.

3) Set securityParameters to a zero-length OCTET STRING ('0400').
4) Combine the message parts into a wholeMsg and calculate wholeMsgLength.
5) The wholeMsg, wholeMsgLength, securityParameters and tmStateReference are returned to the calling Message Processing Model with the statusInformation set to success.

---

5. Processing an Incoming SNMP Message

TOC

An error indication might return an OID and value for an incremented counter and a value for securityLevel, and values for contextEngineID

and contextName for the counter, and the securityStateReference if the information is available at the point where the error is detected.

5.1. Security Processing for an Incoming Message

TOC

This section describes the procedure followed by the Transport Security Model whenever it receives an incoming message from a Message Processing Model. The ASI from a Message Processing Model to the Security Subsystem for a received message is:

```
statusInformation =  -- errorIndication or success
                         -- error counter OID/value if error
processIncomingMsg(
IN   messageProcessingModel    -- typically, SNMP version
IN   maxMessageSize            -- from the received message
IN   securityParameters        -- from the received message
IN   securityModel             -- from the received message
IN   securityLevel             -- from the received message
IN   wholeMsg                  -- as received on the wire
IN   wholeMsgLength            -- length as received on the wire
IN   tmStateReference          -- (NEW) from the Transport Model
OUT  securityEngineID          -- authoritative SNMP entity
OUT  securityName              -- identification of the principal
OUT  scopedPDU,                -- message (plaintext) payload
OUT  maxSizeResponseScopedPDU  -- maximum size sender can handle
OUT  securityStateReference    -- reference to security state
 )                          -- information, needed for response
```

5.2. Elements of Procedure for Incoming Messages

TOC

1) Set the securityEngineID to the local snmpEngineID.
2) If tmStateReference does not refer to a cache containing values for tmTransportDomain, tmTransportAddress, tmSecurityName and tmTransportSecurityLevel, then the snmpTsmInvalidCaches counter is incremented, an error indication is returned to the calling module, and Security Model processing stops for this message.
3) Copy the tmSecurityName to securityName.
If the snmpTsmConfigurationUsePrefix object is set to true, then use the tmTransportDomain to look up the corresponding prefix.

> If the prefix lookup fails for any reason, then the snmpTsmUnknownPrefixes counter is incremented, an error indication is returned to the calling module, and message processing stops.

> If the lookup succeeds, but the prefix length is less than one or greater than four octets, then the snmpTsmInvalidPrefixes counter is incremented, an error indication is returned to the calling module, and message processing stops.
>
> Set the securityName to be the concatenation of the prefix, a ':' character (US-ASCII 0x3a) and the tmSecurityName.

4) Compare the value of tmTransportSecurityLevel in the tmStateReference cache to the value of the securityLevel parameter passed in the processIncomingMsg ASI. If securityLevel specifies privacy (Priv), and tmTransportSecurityLevel specifies no privacy (noPriv), or securityLevel specifies authentication (auth) and tmTransportSecurityLevel specifies no authentication (noAuth) was provided by the Transport Model, then the snmpTsmInadequateSecurityLevels counter is incremented, an error indication (unsupportedSecurityLevel) together with the OID and value of the incremented counter is returned to the calling module, and Transport Security Model processing stops for this message.
5) The tmStateReference is cached as cachedSecurityData, so that a possible response to this message will use the same security parameters. Then securityStateReference is set for subsequent reference to this cached data.
6) The scopedPDU component is extracted from the wholeMsg.
7) The maxSizeResponseScopedPDU is calculated. This is the maximum size allowed for a scopedPDU for a possible Response message.
8) The statusInformation is set to success and a return is made to the calling module passing back the OUT parameters as specified in the processIncomingMsg ASI.

---

6. MIB Module Overview

TOC

This MIB module provides objects for use only by the Transport Security Model. It defines a configuration scalar and related error counters.

6.1. Structure of the MIB Module

TOC

Objects in this MIB module are arranged into subtrees. Each subtree is organized as a set of related objects. The overall structure and assignment of objects to their subtrees, and the intended purpose of each subtree, is shown below.

6.1.1. The snmpTsmStats Subtree

TOC

This subtree contains error counters specific to the Transport Security Model.

6.1.2. The snmpTsmConfiguration Subtree

TOC

This subtree contains a configuration object that enables administrators to specify if they want a transport domain prefix prepended to securityNames for use by applications.

6.2. Relationship to Other MIB Modules

TOC

Some management objects defined in other MIB modules are applicable to an entity implementing the Transport Security Model. In particular, it is assumed that an entity implementing the Transport Security Model will implement the SNMP-FRAMEWORK-MIB [RFC3411] (Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks," December 2002.), the SNMP-TARGET-MIB [RFC3413] (Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications," December 2002.), the SNMP-VIEW-BASED-ACM-MIB [RFC3415] (Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)," December 2002.), and the SNMPv2-MIB [RFC3418] (Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)," December 2002.). These are not needed to implement the SNMP-TSM-MIB.

6.2.1. MIB Modules Required for IMPORTS

TOC

The following MIB module imports items from [RFC2578] (McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)," April 1999.), [RFC2579] (McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Textual Conventions for SMIv2," April 1999.), and [RFC2580] (McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Conformance Statements for SMIv2," April 1999.).

---

7. MIB module definition

TOC

```
SNMP-TSM-MIB DEFINITIONS ::= BEGIN

IMPORTS
    MODULE-IDENTITY, OBJECT-TYPE,
    mib-2, Counter32
      FROM SNMPv2-SMI -- RFC2578
    MODULE-COMPLIANCE, OBJECT-GROUP
      FROM SNMPv2-CONF -- RFC2580
    TruthValue
       FROM SNMPv2-TC -- RFC2579
    ;

snmpTsmMIB MODULE-IDENTITY
    LAST-UPDATED "200903090000Z"
    ORGANIZATION "ISMS Working Group"
    CONTACT-INFO "WG-EMail:   isms@lists.ietf.org
                  Subscribe:  isms-request@lists.ietf.org

                  Chairs:
                    Juergen Quittek
                    NEC Europe Ltd.
                    Network Laboratories
                    Kurfuersten-Anlage 36
                    69115 Heidelberg
                    Germany
                    +49 6221 90511-15
                    quittek@netlab.nec.de

                    Juergen Schoenwaelder
                    Jacobs University Bremen
                    Campus Ring 1
                    28725 Bremen
                    Germany
                    +49 421 200-3587
                    j.schoenwaelder@iu-bremen.de

                  Editor:
                    David Harrington
                    Huawei Technologies USA
                    1700 Alma Dr.
                    Plano TX 75075
                    USA
                    +1 603-436-8634
                    ietfdbh@comcast.net

                    Wes Hardaker
                    Sparta, Inc.
                    P.O. Box 382
```

```
                Davis, CA  95617
                USA
                +1 530 792 1913
                ietf@hardakers.net
              "
 DESCRIPTION "The Transport Security Model MIB

              In keeping with the RFC 3411 design decisions
              to use self-contained documents, the RFC which
              contains the definition of this MIB module also
              includes the elements of procedure which are
              needed for processing the Transport Security
              Model for SNMP. These MIB objects
              SHOULD NOT be modified via other subsystems
              or models defined in other documents.
              This allows the Transport Security Model
              for SNMP to be designed and documented as
              independent and self- contained, having no
              direct impact on other modules, and this
              allows this module to be upgraded and
              supplemented as the need arises, and to
              move along the standards track on different
              time-lines from other modules.


```

```
    USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED
    AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
    LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN
    ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
    POSSIBILITY OF SUCH DAMAGE.

    This version of this MIB module is part of RFC XXXX;
    see the RFC itself for full legal notices.
-- NOTE to RFC editor: replace XXXX with actual RFC number
--                     for this document and remove this note
                "

    REVISION    "200903090000Z"
    DESCRIPTION "The initial version, published in RFC XXXX.
-- NOTE to RFC editor: replace XXXX with actual RFC number
--                     for this document and remove this note
                "

    ::= { mib-2 xxxx }
-- RFC Ed.: replace xxxx with IANA-assigned number and
--          remove this note

-- ------------------------------------------------------------ --
-- subtrees in the SNMP-TSM-MIB
-- ------------------------------------------------------------ --

snmpTsmNotifications OBJECT IDENTIFIER ::= { snmpTsmMIB 0 }
snmpTsmMIBObjects    OBJECT IDENTIFIER ::= { snmpTsmMIB 1 }
snmpTsmConformance   OBJECT IDENTIFIER ::= { snmpTsmMIB 2 }

-- ------------------------------------------------------------
-- Objects
-- ------------------------------------------------------------

-- Statistics for the Transport Security Model


snmpTsmStats         OBJECT IDENTIFIER ::= { snmpTsmMIBObjects 1 }

snmpTsmInvalidCaches OBJECT-TYPE
    SYNTAX       Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The number of incoming messages dropped because the
                 tmStateReference referred to an invalid cache.
                "
    ::= { snmpTsmStats 1 }

snmpTsmInadequateSecurityLevels OBJECT-TYPE
    SYNTAX       Counter32
```

```
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The number of incoming messages dropped because
                 the securityLevel asserted by the transport model was
                 less than the securityLevel requested by the
                 application.
                "
    ::= { snmpTsmStats 2 }

snmpTsmUnknownPrefixes OBJECT-TYPE
    SYNTAX       Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The number of messages dropped because
                 snmpTsmConfigurationUsePrefix was set to true and
                 there is no known prefix for the specified transport
                 domain.
                "
    ::= { snmpTsmStats 3 }

snmpTsmInvalidPrefixes OBJECT-TYPE
    SYNTAX       Counter32
    MAX-ACCESS   read-only
    STATUS       current
    DESCRIPTION "The number of messages dropped because
                 the securityName associated with an outgoing message
                 did not contain a valid transport domain prefix.
                "
    ::= { snmpTsmStats 4 }

-- ------------------------------------------------------------
-- Configuration
-- ------------------------------------------------------------

-- Configuration for the Transport Security Model


snmpTsmConfiguration   OBJECT IDENTIFIER ::= { snmpTsmMIBObjects 2 }

snmpTsmConfigurationUsePrefix OBJECT-TYPE
    SYNTAX      TruthValue
    MAX-ACCESS  read-write
    STATUS      current
    DESCRIPTION "If this object is set to true then securityNames
                 passing to and from the application are expected to
                 contain a transport domain specific prefix. If this
                 object is set to true then a domain specific prefix
                 will be added by the TSM to the securityName for
                 incoming messages and removed from the securityName
```

```
                when processing outgoing messages. Transport domains
                and prefixes are maintained in a registry by IANA.
                This object SHOULD persist across system reboots.
               "
    DEFVAL { false }
    ::= { snmpTsmConfiguration 1 }

-- ----------------------------------------------------------
-- snmpTsmMIB - Conformance Information
-- ----------------------------------------------------------

snmpTsmCompliances OBJECT IDENTIFIER ::= { snmpTsmConformance 1 }

snmpTsmGroups      OBJECT IDENTIFIER ::= { snmpTsmConformance 2 }

-- ----------------------------------------------------------
-- Compliance statements
-- ----------------------------------------------------------

snmpTsmCompliance MODULE-COMPLIANCE
    STATUS      current
    DESCRIPTION "The compliance statement for SNMP engines that support
                 the SNMP-TSM-MIB
                "
    MODULE
        MANDATORY-GROUPS { snmpTsmGroup }
    ::= { snmpTsmCompliances 1 }

-- ----------------------------------------------------------
-- Units of conformance
-- ----------------------------------------------------------
snmpTsmGroup OBJECT-GROUP
    OBJECTS {
        snmpTsmInvalidCaches,
        snmpTsmInadequateSecurityLevels,
        snmpTsmUnknownPrefixes,
        snmpTsmInvalidPrefixes,
        snmpTsmConfigurationUsePrefix
    }
    STATUS      current
    DESCRIPTION "A collection of objects for maintaining
                 information of an SNMP engine which implements
                 the SNMP Transport Security Model.
                "

    ::= { snmpTsmGroups 2 }


END
```

8. Security Considerations

TOC

This document describes a Security Model, compatible with the RFC3411 architecture, that permits SNMP to utilize security services provided through an SNMP Transport Model. The Transport Security Model relies on Transport Models for mutual authentication, binding of keys, confidentiality and integrity.
The Transport Security Model relies on secure Transport Models to provide an authenticated principal identifier and an assertion of whether authentication and privacy are used during transport. This Security Model SHOULD always be used with Transport Models that provide adequate security, but "adequate security" is a configuration and/or run-time decision of the operator or management application. The security threats and how these threats are mitigated should be covered in detail in the specifications of the Transport Models and the underlying secure transports.
An authenticated principal identifier (securityName) is used in SNMP applications, for purposes such as access control, notification generation, and proxy forwarding. This security model supports multiple transport models. Operators might judge some transports to be more secure than others, so this security model can be configured to prepend a prefix to the securityName to indicate the transport model used to authenticate the principal. Operators can use the prefixed securityName when making application decisions about levels of access.

8.1. MIB module security

TOC

There are a number of management objects defined in this MIB module with a MAX-ACCESS clause of read-write and/or read-create. Such objects may be considered sensitive or vulnerable in some network environments. The support for SET operations in a non-secure environment without proper protection can have a negative effect on network operations. These are the tables and objects and their sensitivity/vulnerability:

- The snmpTsmConfigurationUsePrefix object could be modified, creating a denial of service or authorizing SNMP messages that would not have previously been authorized by an Access Control Model (e.g. the VACM).

Some of the readable objects in this MIB module (i.e., objects with a MAX-ACCESS other than not-accessible) may be considered sensitive or vulnerable in some network environments. It is thus important to control even GET and/or NOTIFY access to these objects and possibly to

even encrypt the values of these objects when sending them over the network via SNMP. These are the tables and objects and their sensitivity/vulnerability:

- *All the counters in this module refer to configuration errors and do not expose sensitive information.

SNMP versions prior to SNMPv3 did not include adequate security. Even if the network itself is secure (for example by using IPsec), even then, there is no control as to who on the secure network is allowed to access and GET/SET (read/change/create/delete) the objects in this MIB module.
It is RECOMMENDED that implementers consider the security features as provided by the SNMPv3 framework (see [RFC3410] (Case, J., Mundy, R., Partain, D., and B. Stewart, “Introduction and Applicability Statements for Internet-Standard Management Framework,” December 2002.) section 8), including full support for the USM and Transport Security Model cryptographic mechanisms (for authentication and privacy).
Further, deployment of SNMP versions prior to SNMPv3 is NOT RECOMMENDED. Instead, it is RECOMMENDED to deploy SNMPv3 and to enable cryptographic security. It is then a customer/operator responsibility to ensure that the SNMP entity giving access to an instance of this MIB module is properly configured to give access to the objects only to those principals (users) that have legitimate rights to indeed GET or SET (change/create/delete) them.

---

9. IANA Considerations

TOC

IANA is requested to assign:

1. an SMI number with a prefix of mib-2, in the MIB module registry under http://www.iana.org/assignments/smi-numbers, for the MIB module in this document,

2. a value, preferably 4, to identify the Transport Security Model, in the Security Models registry at http://www.iana.org/assignments/snmp-number-spaces. This should result in the following table of values:

```
Value   Description                          References
-----   -----------                          ----------
  0     reserved for 'any'                   [RFC3411]
  1     reserved for SNMPv1                  [RFC3411]
  2     reserved for SNMPv2c                 [RFC3411]
  3     User-Based Security Model (USM)      [RFC3411]
  YY    Transport Security Model (TSM)       [RFCXXXX]

-- NOTE to RFC editor: replace XXXX with actual RFC number
--                     for this document and remove this note
-- NOTE to RFC editor: replace YY with actual IANA-assigned number,
                       throughout this document and remove this note.
```

---

10. Acknowledgements

TOC

The editors would like to thank Jeffrey Hutzelman for sharing his SSH insights, and Dave Shield for an outstanding job wordsmithing the existing document to improve organization and clarity.
Additionally, helpful document reviews were received from: Juergen Schoenwaelder.

---

11. References

TOC

11.1. Normative References

TOC

[RFC2119]	Bradner, S., “Key words for use in RFCs to Indicate Requirement Levels,” BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).
[RFC2578]	McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., “Structure of Management Information Version 2 (SMIv2),” STD 58, RFC 2578, April 1999 (TXT).
[RFC2579]	McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., “Textual Conventions for SMIv2,” STD 58, RFC 2579, April 1999 (TXT).
[RFC2580]	McCloghrie, K., Perkins, D., and J. Schoenwaelder, “Conformance Statements for SMIv2,” STD 58, RFC 2580, April 1999 (TXT).
[RFC3411]	

	Harrington, D., Presuhn, R., and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks," STD 62, RFC 3411, December 2002 (TXT).
[RFC3412]	Case, J., Harrington, D., Presuhn, R., and B. Wijnen, "Message Processing and Dispatching for the Simple Network Management Protocol (SNMP)," STD 62, RFC 3412, December 2002 (TXT).
[RFC3413]	Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications," STD 62, RFC 3413, December 2002 (TXT).
[RFC3414]	Blumenthal, U. and B. Wijnen, "User-based Security Model (USM) for version 3 of the Simple Network Management Protocol (SNMPv3)," STD 62, RFC 3414, December 2002 (TXT).
[I-D.ietf-isms-tmsm]	Harrington, D. and J. Schoenwaelder, "Transport Subsystem for the Simple Network Management Protocol (SNMP)," draft-ietf-isms-tmsm-18 (work in progress), May 2009 (TXT).

11.2. Informative References

TOC

[RFC3410]	Case, J., Mundy, R., Partain, D., and B. Stewart, "Introduction and Applicability Statements for Internet-Standard Management Framework," RFC 3410, December 2002 (TXT).
[RFC3415]	Wijnen, B., Presuhn, R., and K. McCloghrie, "View-based Access Control Model (VACM) for the Simple Network Management Protocol (SNMP)," STD 62, RFC 3415, December 2002 (TXT).
[RFC3418]	Presuhn, R., "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)," STD 62, RFC 3418, December 2002 (TXT).
[RFC3584]	Frye, R., Levi, D., Routhier, S., and B. Wijnen, "Coexistence between Version 1, Version 2, and Version 3 of the Internet-standard Network Management Framework," BCP 74, RFC 3584, August 2003 (TXT).
[I-D.ietf-isms-secshell]	Harrington, D., Salowey, J., and W. Hardaker, "Secure Shell Transport Model for SNMP," draft-ietf-isms-secshell-18 (work in progress), May 2009 (TXT).

TOC

Appendix A. Notification Tables Configuration

The SNMP-TARGET-MIB and SNMP-NOTIFICATION-MIB [RFC3413] (Levi, D., Meyer, P., and B. Stewart, "Simple Network Management Protocol (SNMP) Applications," December 2002.) are used to configure notification originators with the destinations to which notifications should be sent.
Most of the configuration is security-model-independent and transport-model-independent.
The values we will use in the examples for the five model-independent security and transport parameters are:

```
transportDomain = snmpSSHDomain

transportAddress = 192.0.2.1:PPP

securityModel = Transport Security Model

securityName = alice

securityLevel = authPriv
```

[-- NOTE to RFC editor: replace PPP above with actual IANA-assigned port number for SNMP notifications over SSH, from draft-ietf-isms-secshell, and remove this note.]
The following example will configure the Notification Originator to send informs to a Notification Receiver at 192.0.2.1:PPP using the securityName "alice". "alice" is the name for the recipient from the standpoint of the notification originator, and is used for processing access controls before sending a notification.
[-- NOTE to RFC editor: replace PPP above with actual IANA-assigned port number for SNMP notifications over SSH, and remove this note.]
The columns marked with a "*" are the items that are Security Model or Transport Model specific.
The configuration for the "alice" settings in the SNMP-VIEW-BASED-ACM-MIB objects are not shown here for brevity. First we configure which type of notification will be sent for this taglist (toCRTag). In this example, we choose to send an Inform.

```
       snmpNotifyTable row:
            snmpNotifyName                CRNotif
            snmpNotifyTag                 toCRTag
            snmpNotifyType                inform
            snmpNotifyStorageType         nonVolatile
            snmpNotifyColumnStatus        createAndGo
```

Then we configure a transport address to which notifications associated with this taglist will be sent, and we specify which snmpTargetParamsEntry will be used (toCR) when sending to this transport address.

```
 snmpTargetAddrTable row:
    snmpTargetAddrName              toCRAddr
*   snmpTargetAddrTDomain           snmpSSHDomain
*   snmpTargetAddrTAddress          192.0.2.1:PPP
    snmpTargetAddrTimeout           1500
    snmpTargetAddrRetryCount        3
    snmpTargetAddrTagList           toCRTag
    snmpTargetAddrParams            toCR   (MUST match below)
    snmpTargetAddrStorageType       nonVolatile
    snmpTargetAddrColumnStatus      createAndGo
```

[-- NOTE to RFC editor: replace PPP above with actual IANA-assigned port number for SNMP notifications over SSH, and remove this note.] Then we configure which principal at the host will receive the notifications associated with this taglist. Here we choose "alice", who uses the Transport Security Model.

```
snmpTargetParamsTable row:
    snmpTargetParamsName            toCR
    snmpTargetParamsMPModel         SNMPv3
*   snmpTargetParamsSecurityModel   TransportSecurityModel
    snmpTargetParamsSecurityName    "alice"
    snmpTargetParamsSecurityLevel   authPriv
    snmpTargetParamsStorageType     nonVolatile
    snmpTargetParamsRowStatus       createAndGo
```

A.1. Transport Security Model Processing for Notifications

TOC

The Transport Security Model is called using the generateRequestMsg() ASI, with the following parameters (* are from the above tables):

```
statusInformation =                -- success or errorIndication
      generateRequestMsg(
      IN   messageProcessingModel  -- *snmpTargetParamsMPModel
      IN   globalData              -- message header, admin data
      IN   maxMessageSize          -- of the sending SNMP entity
      IN   transportDomain         -- *snmpTargetAddrTDomain
      IN   transportAddress        -- *snmpTargetAddrTAddress
      IN   securityModel           -- *snmpTargetParamsSecurityModel
      IN   securityEngineID        -- immaterial; TSM will ignore.
      IN   securityName            -- snmpTargetParamsSecurityName
      IN   securityLevel           -- *snmpTargetParamsSecurityLevel
      IN   scopedPDU               -- message (plaintext) payload
      OUT  securityParameters      -- filled in by Security Module
      OUT  wholeMsg                -- complete generated message
      OUT  wholeMsgLength          -- length of generated message
      OUT  tmStateReference        -- reference to transport info
           )
```

The Transport Security Model will determine the Transport Model based on the snmpTargetAddrTDomain. The selected Transport Model will select the appropriate transport connection using the tmStateReference cache created from the values of snmpTargetAddrTAddress, snmpTargetParamsSecurityName, and snmpTargetParamsSecurityLevel.

Appendix B. Processing Differences between USM and Secure Transport

TOC

USM and secure transports differ in the processing order and responsibilities within the RFC3411 architecture. While the steps are the same, they occur in a different order, and might be done by different subsystems. The following lists illustrate the difference in the flow and the responsibility for different processing steps for incoming messages when using USM and when using a secure transport. (These lists are simplified for illustrative purposes, and do not represent all details of processing. Transport Models MUST provide the detailed elements of procedure.)

With USM, SNMPv1, and SNMPv2c Security Models, security processing starts when the Message Processing Model decodes portions of the ASN.1 message to extract header fields that are used to determine which Security Model will process the message to perform authentication, decryption, timeliness checking, integrity checking, and translation of parameters to model-independent parameters. By comparison, a secure transport performs those security functions on the message, before the ASN.1 is decoded.

Step 6 cannot occur until after decryption occurs. Step 6 and beyond are the same for USM and a secure transport.

B.1. USM and the RFC3411 Architecture

TOC

1) decode the ASN.1 header (Message Processing Model)

2) determine the SNMP Security Model and parameters (Message Processing Model)

3) verify securityLevel. [Security Model]

4) translate parameters to model-independent parameters (Security Model)

5) authenticate the principal, check message integrity and timeliness, and decrypt the message. [Security Model]

6) determine the pduType in the decrypted portions (Message Processing Model), and

7) pass on the decrypted portions with model-independent parameters.

B.2. Transport Subsystem and the RFC3411 Architecture

TOC

1) authenticate the principal, check integrity and timeliness of the message, and decrypt the message. [Transport Model]

2) translate parameters to model-independent parameters (Transport Model)

3) decode the ASN.1 header (Message Processing Model)

4) determine the SNMP Security Model and parameters (Message Processing Model)

5) verify securityLevel [Security Model]

6) determine the pduType in the decrypted portions (Message Processing Model), and

7) pass on the decrypted portions with model-independent security parameters

If a message is secured using a secure transport layer, then the Transport Model will provide the translation from the authenticated identity (e.g., an SSH user name) to a human-friendly identifier (tmSecurityName) in step 2. The security model will provide a mapping from that identifier to a model-independent securityName.

Authors' Addresses

TOC

	David Harrington
	Huawei Technologies (USA)
	1700 Alma Dr. Suite 100
	Plano, TX 75075
	USA
Phone:	+1 603 436 8634
EMail:	dharrington@huawei.com
	Wes Hardaker
	Sparta, Inc.
	P.O. Box 382
	Davis, CA 95617
	US
Phone:	+1 530 792 1913
EMail:	ietf@hardakers.net